
Scalable Computing: Practice and Experience, ISSN 1895-1767, http://www.scpe.org
© 2020 SCPE. Volume 21, Issue 2, pp. 159–172, DOI 10.12694:/scpe.v21i2.1670

CPU-MEMORY AWARE VM CONSOLIDATION FOR CLOUD DATA CENTERS

B. NITHIYA *AND R. ESWARI †

Abstract. The unbalanced usage of resources in cloud data centers cause an enormous amount of power consumption. The Virtual Machine (VM) consolidation shuts the underutilized hosts and makes the overloaded hosts as normally loaded hosts by selecting appropriate VMs from the hosts and migrates them to other hosts in such a way to reduce the energy consumption and to improve physical resource utilization. Efficient method is needed for VM selection and destination hosts selection (VM placement). In this paper, a CPU-Memory aware VM placement algorithm is proposed for selecting suitable destination host for migration. The VMs are selected using Fuzzy Soft Set (FSS) method VM selection algorithm. The proposed placement algorithm considers both CPU, Memory, and combination of CPU-Memory utilization of VMs on the source host. The proposed method is experimentally compared with several existing selection and placement algorithms and the results show that the proposed consolidation method performs better than existing algorithms in terms of energy efficiency, energy consumption, SLA violation rate, and number of VM migrations.

Key words: cloud computing, VM consolidation, VM Placement, Energy Consumption, Energy Efficiency, SLA Violation Rate.

AMS subject classifications. 68M14

1. Introduction. Due to the increase of storage demands the cloud service providers launched the cloud data centers for satisfying the user needs. Various users' VM requests are complex and required taking into account of multiple resource constraints. So, the physical servers that satisfy the users request will be considered to deploy the selected VMs [1]. Commercial IaaS cloud companies, including Amazon EC2 [2], IBM [3] and Google Compute Engine [4] are offering various types of VM instances with varying types and resource volumes. As a significance, when cloud data center owners are unable to deploy different types of VM requests efficiently, some assets may get overloaded while others remain underutilized. These unbalanced resource usages can eventually lead to unnecessary physical server activation. Thus, a significant concern is needed to balance the load in terms of CPU, memory, storage, and network bandwidth while meeting all requests of VM.

Consolidation is the process of moving running VM from one physical server to another without down time and switch off idle servers to power save mode. There are two types of consolidation: Static where VMs size is fixed; Dynamic where periodical demands in the each VM. Dynamic consolidation is performed in two steps one is migrate VMs from underutilized host and put the host in sleep mode. Another step is to migrate VMs from overloaded host without degrading the performance such as energy consumption, SLA violation rate [5].

For dynamic VM consolidation, there are several VM selection and VM placement algorithms have been proposed for selecting VMs from overloaded host and placing them on to the appropriate destination host. In this paper, dynamic consolidation is considered. In this paper a fuzzy soft set based VM selection algorithm [6] is used for selecting VMs from host which considers all factors such as RAM, CPU, memory, and correlation values. A CPU-Memory aware placement algorithm is proposed which considers both CPU and memory factors for selecting appropriate hosts for deploying selected VMs for migration. The abbreviations are used in this paper are listed in Table 1.1.

The remainder of the paper is structured as follows: Section 2 presents the related works. Section 3 describes the VM consolidation and the proposed CPU-Memory aware VM placement algorithm. In Section 4,

*Department of Computer Applications, National Institute of Technology, Tiruchirappalli, Tamilnadu, India (nithiyab1988@gmail.com).

†Department of Computer Applications, National Institute of Technology, Tiruchirappalli, Tamilnadu, India (eswari@nitt.edu).

Table 1.1
Abbreviation

Abbreviation	Explanation
VM	Virtual Machine
FSS	Fuzzy Soft Set
MEM	Memory Utilization
RCM	Ratio of CPU utilization to Memory utilization
PCM	Product of CPU utilization to Memory utilization
MCC	Minimum Correlation Coefficient
IQR	Inter-Quartile Range
LR	Local Regression
LRR	Local Robust Regression
MAD	Median Absolute Deviation
THR	Threshold
RS	Random Solution
MU	Minimum Utilization
MC	Maximum Correlation
MP	Meet Performance
AFT-FS	Adaptive Four Threshold based Fuzzy VM Selection
PABFD	Power-Aware Best-Fit Decreasing
EC	Energy Consumption
SLA violation rate	Service Level Agreement violation rate
SVTAH	SLA Violation Time per Active Host
PDCVM	Performance Degradation Caused by VM Migration
ESV	Product of Energy and SLA Violation
EE	Energy Efficiency

the experimental setup and evaluation metrics are discussed. In Section 5, the experimental results of existing and proposed methods are discussed. Finally, Section 6 concludes the work with future extension.

2. Related Work. The dynamic VM consolidation is done by 3 levels: Host classification, VM Selection, and VM Placement. Several Selection, and Placement algorithms have been proposed by researchers.

2.1. Host Classification. For host allocation the following algorithms are used by researchers [5]: Inter Quartile Range (IQR), Local Regression (LR), Local Robust Regression (LRR), Median Absolute Deviation (MAD), and Static Threshold (THR). The authors [5] found that THR is the best VM allocation algorithm than others.

An adaptive four threshold method [7] is used to classify the hosts. The thresholds are determined using K-means clustering midrange inter quartile range algorithm [8].

2.2. VM Selection. The Minimum Migration Time (MMT) algorithm [5] migrates a VM that needs minimum migration time to complete the migration from overloaded host to less loaded host. The migration time could be calculated as the ratio of amount of RAM utilized by VM to the network bandwidth available for the host.

Higher the usage of resources in the server by the applications, greater the chance of the server getting overloaded. So, the authors in [5] proposed maximum correlation algorithm to select the VMs that have higher correlation of the CPU utilization compared with other VMs for migration. The multiple correlation coefficient is applied to evaluate the correlation between CPU utilization of VMs.

In Minimum Utilization (MU) algorithm [5], the VM that uses high CPU will not be considered for migration since its migration increases downtime (the period during which the service is unavailable due to there being no currently executing instance of that VM). So, the VMs that have minimum CPU utilization is selected for migration. The Random Selection (RS) algorithm [5] selects VMs randomly without any rules.

Meet Performance (MP) selection algorithm was proposed by [10] virtual which may vary from the other

selection algorithms. Their algorithm compares host's utilization deviation with upper threshold and with CPU utilization of VMs in the host. The selection of VMs is based on the comparison results. The VM that has the lowest resource satisfaction will get higher priority to be migrated.

An adaptive fuzzy based VM selection algorithm (AFT_FS) was proposed by us [7] which uses four threshold values to detect overloaded hosts and a fuzzy-based approach to select VMs for migration.

All of the above mentioned selection algorithms have considered any one of the selection factors such as either RAM or CPU or Memory or Correlation values for selecting VMs during migration.

A Fuzzy Soft Set (FSS) based VM Selection algorithm was proposed by us [9] to achieve the optimal selection of VMs for migration. The algorithm considers all four factors at a time, and accurately finds which VM has to migrate from the overloaded hosts in the cloud data center.

2.3. VM Placement. Virtual Machine Placement is a crucial issue in cloud computing. It is a method where most appropriate physical machine (PM) will be selected to place VMs. The VM placement also plays important role during dynamic VM consolidation. Most of the researchers concentrate on initial VM placement. Many heuristic and meta-heuristic algorithms have been proposed for initial VM placement such as Ant Colony System (ACS) embedded with First Fit Decreasing (FFD) [11], Grey Wolf Optimization (GWO) [12], Genetic Algorithm [13] etc. But our research work mainly focuses on the VM placement during VM consolidation. Since the meta-heuristic algorithms consume more time to take the decision for PM-VM pair during consolidation, our research work considers only heuristic algorithms to select appropriate destination host.

PABFD algorithm [5] is most commonly used to place VMs onto the destination hosts. It sorts all the VMs in the decreasing order of their CPU utilization values and allocates each VM to a host that provides the minimum raise of energy consumption due to this allocation.

Most of the VM consolidation methods use this algorithm to select the hosts to which the VMs can be placed. This placement algorithm considers only the CPU utilization of hosts during host selection.

The Minimum Correlation Coefficient (MCC) algorithm was proposed by [10]. It finds the correlation coefficient between chosen VM and target host based on the utilization of CPU alone. The authors place the chosen VM to the host that has the minimum correlation coefficient with it.

This research work tries to investigate the impact of memory utilization, CPU utilization, and the combination of both for selecting the target host. The proposed algorithm finds the minimum correlation coefficient between chosen VM and target host based on memory, CPU, Ratio of memory to CPU, and Product of memory and CPU.

3. Proposed Work. In this paper, the CPU-Memory aware VM placement algorithm is proposed for cloud data centers. The algorithm considers three different utilization matrices based on CPU and Memory utilization of VMs on targeted hosts during p time slices. The association (correlation) between VM and host will be separately calculated for each resource (utilization matrix). The host that gives minimum correlation will be selected for placement of VM. The nomenclature used in this paper are listed in Table 1. and the overall flow chart of VM Consolidation is shown in Fig 3.1.

3.1. VM Consolidation. VM Consolidation is used to maintain the balance between energy and QoS. An efficient VM consolidation method should minimize energy consumption, SLA violation rate, and maximize energy efficiency. It should also have efficient VM migration, and minimum number of active hosts at a given time. Fig 3.2 shown that the host classification of datacenter.

It considers the following steps as given below.

- All hosts in the data centers are clustered into 5 groups using K-Means Inter Quartile Range clustering algorithm [7]: overloaded hosts, normally loaded hosts, little loaded hosts, less loaded hosts, and idle hosts.
- Migrate VMs if any on idle hosts to less loaded hosts and move hosts to power save mode.
- Migrate all VMs from little loaded hosts to normally loaded host. Then move all little loaded hosts to power save mode.
- Select VMs from the overloaded hosts using FSS algorithm and migrate them to less loaded hosts.

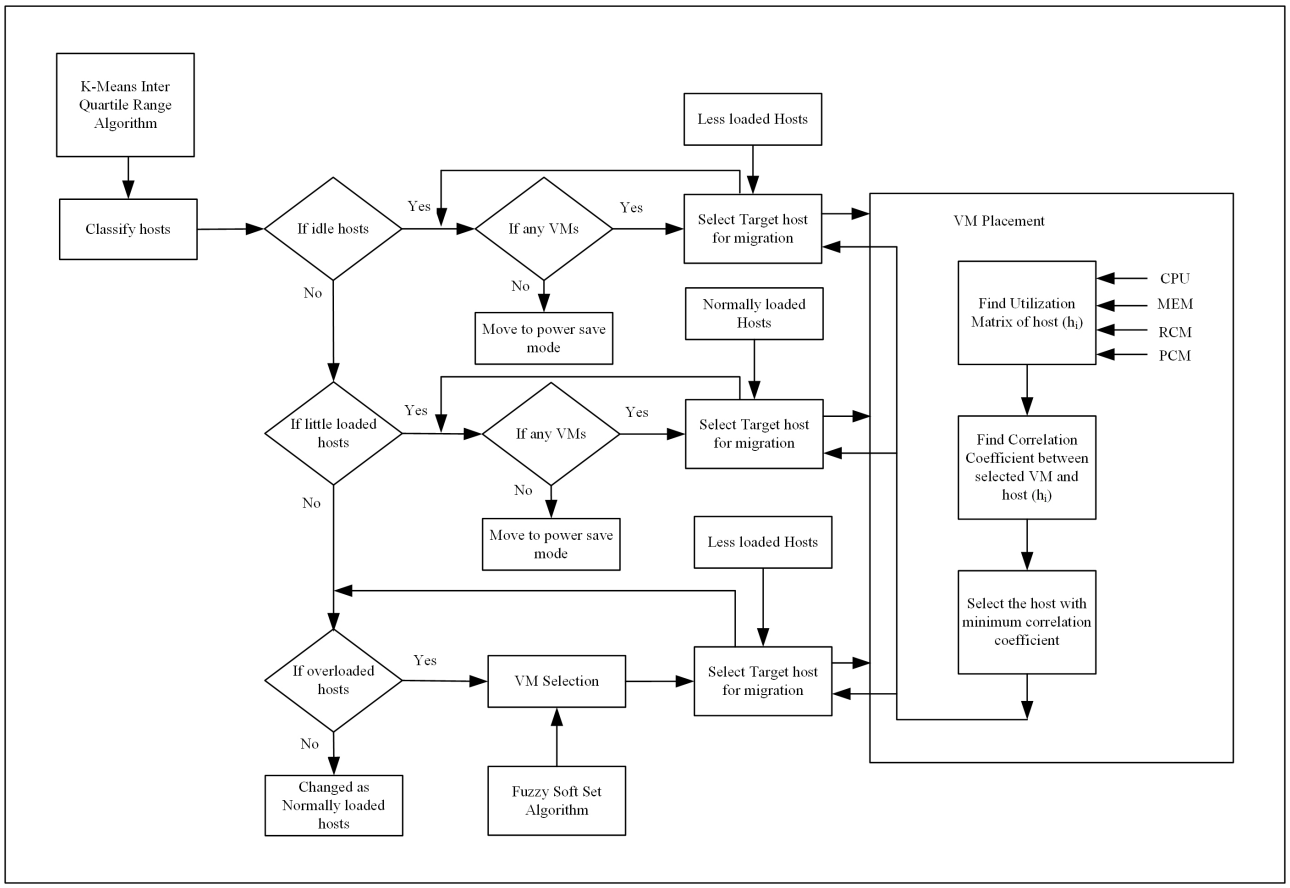

FIG. 3.1. *Flow Chart of VM Consolidation*

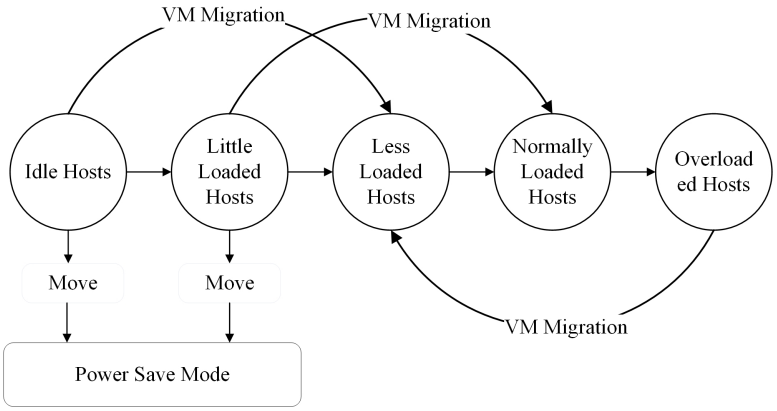

FIG. 3.2. *Host Classification*

3.2. CPU-Memory aware VM Placement Algorithm. Once a VM is selected using FSS for migration, a VM placement algorithm will be used to select an appropriate host for migration. The placement algorithm must minimize power consumption, and SLA violation rate.

3.2.1. VM Placement. The proposed algorithm consolidates VMs using three different ways of placement algorithm. The first one is based on memory utilization of selected VMs and second and third are based on

Algorithm 1 VM_Consolidation

Require: Hosts in datacenter, VM_list
Ensure: Select the Target_host

```
1: Cluster all hosts into five: Overloaded_hosts, Normallyloaded_hosts, Lessloaded_hosts, Littleloaded_hosts, Idle_hosts based on threshold values
2: Placement_Policy ← Get Placement_Policy
3: if (Idle_hosts) then
4:     for each host h_i in Idle_hosts do
5:         for each VM v in host h_i do
6:             Target_host ← Select_host (v, Lessloaded_hosts, Placement_Policy)
7:             Migrate v to Target_host
8:         Move host h_i to power save mode
9: else if (Littleloaded_hosts) then
10:    for each host h_i in Littleloaded_hosts do
11:        for each VM v in host h_i do
12:            Target_host ← Select_host (v, Normallyloaded_hosts, Placement_Policy)
13:            Migrate v to Target_host
14:        Move host h_i to power save mode
15: else(Overloaded_hosts)
16:    for each host h_i in Overloaded_hosts do
17:        repeat
18:            Select VM_Migrate list in h_i using fuzzy soft set
19:            for each VM v in VM_Migrate list do
20:                Target_host ← Select_host (v, Lessloaded_hosts, Placement_Policy)
21:                Migrate v to Target_host
22:        until host h_i becomes Normallyloaded
```

CPU and memory utilization of selected VMs for every p time slice.

VM Consolidation is given in algorithm 1. Step 1 clusters the hosts into five: Overloaded_hosts, Normallyloaded_hosts, Lessloaded_hosts, Littleloaded_hosts, and Idle_hosts based on their threshold values. Step 2 gets the Placement_Policy. Steps 3 to 8: Select target hosts from Lessloaded_hosts to place all VMs from idle host. This host selection will be repeated for all idle hosts. Now idle hosts are moved to power save mode. Steps 9 to 14: Select target hosts from Normallyloaded_hosts to place all VMs from Littleloaded_hosts. This host selection will be repeated for all little loaded hosts. Now Littleloaded_hosts are moved to power save mode. Steps 15 to 22: Select target host from Lessloaded_hosts for every VM in the overloaded hosts for migration. This host selection will be repeated for all overloaded hosts.

3.2.2. Target Host Selection. Algorithm 2 selects the host to which the chosen VM to be migrated. It uses the Minimum Correlation Coefficient (MCC) [10] to represent the association between chosen VM and target host.

The target host selection algorithm is given in algorithm 2. The algorithm receives the type of hosts from algorithm1. Steps 1 to 3: find the pool of hosts which satisfy the VM v's demand and the total CPU usage of host and VM less than threshold values. Steps 4 to 10: find utilization matrix and squares of correlation coefficient between chosen VM v and all hosts in host pool list. Step 11 selects the target host that has the minimum squared correlation coefficient.

3.2.3. Finding Utilization Matrix. The VM placement is based on three different utilization matrices.

1. Memory Utilization (MEM) Memory utilization method consider only memory resource. For a host h_i with m VMs, the memory utilization of m VMs are collected during p time slices. Now the utilization

Algorithm 2 Select_host (v, Loaded_hosts, Placement_Policy)

Ensure: Select the Target_host

1: **for** each host h_i in Loaded_hosts **do**
2: **if** ((h_i satisfies v's demand) && ((h_i's usage v's demand) <thr)) **then**
3: Host_pool_list ← h_i
4: **for** each host h_i in Host_pool_list **do**
5: Umatrix [m] [p] ← Utilization_Matrix (Placement_Policy, h_i)
6: **for** each time slice k in p **do**
7: Find the resource utilization of h_i

$$sum\_util_i[k] = \sum_{J=1}^{m} Umatrix\_util_i[j][k] \tag{3.1}$$

8: Calculate correlation coefficient between v and h_i

$$\rho_i = \frac{EC[(res\_VM' \frac{1}{p} \sum_{(k=1)}^{p} res\_VM\_util_k)(res\_Host' \frac{1}{p} \sum_{(k=1)}^{p} sum\_util_i[k])]}{\sqrt{(Var(res\_VM))}\sqrt{(Var(res\_Host))}} \tag{3.2}$$

where resource_VM and resource_Host refer to the current resource utilization of the chosen j^{th} VM and the host h_i. $\frac{1}{p}\sum_{(k=1)}^{p} res\_VM\_util_k$ and $\frac{1}{p}\sum_{(k=1)}^{p} sum\_util_i[k])$ refer to the total resource utilization of the chosen j^{th} VM and the host h_i during p time slices. The variance of the resource utilization of the chosen j^{th} VM and host h_i are calculated as

$$Var(res\_VM) = E[(res\_VM - \frac{1}{p}\sum_{(k=1)}^{p} res\_VM\_util_k)^2] \tag{3.3}$$

$$Var(res\_Host) = E[(res\_Host - \frac{1}{p}\sum_{(k=1)}^{p} sum\_util_i[k])^2] \tag{3.4}$$

9: Compute squares of the correlation coefficient

$$\rho = \rho_1^2, \rho_2^2, ', \rho_n^2 \tag{3.5}$$

10: Target_host ← host that has the minimum ρ
11: return Target_host

Algorithm 3 Utilization_Matrix (Placement_Policy, h_i)

Ensure: Utilization_Matrix

1: **if** ($Placement_Policy =' Mem'$) **then**
2: Compute memory utilization matrix using Eqn. 3.6.
3: **else if** ($Placement_Policy =' RCM'$) **then**
4: Compute RCM utilization matrix using Eqn. 3.7.
5: Find RCM values using Eqn. 3.8.
6: **else**($Placement_Policy =' PCM'$)
7: Compute PCM utilization matrix using Eqn. 3.9.
8: Find PCM values using Eqn. 3.10.

matrix of h_i is calculated as the memory utilization of m VMs on h_i at each time slice.

$$Mem\_util_i[m][p] = \begin{bmatrix} Mem_{11} & Mem_{12} & Mem_{13} & \cdots & Mem_{1p} \\ Mem_{21} & Mem_{22} & Mem_{23} & \cdots & Mem_{2p} \\ \vdots & \vdots & Mem_{jk} & \ddots & \vdots \\ Mem_{m1} & Mem_{m2} & Mem_{m3} & \cdots & Mem_{mp} \end{bmatrix} \tag{3.6}$$

where $[RCM]_{jk}$ refers to the ratio of CPU utilization to memory utilization of VM j on host h_i during time slice k

2. Ratio of CPU utilization to Memory utilization Algorithm (RCM) RCM method considers both CPU resource and memory resource. Consider a host h_i and assume that it has m VMs. The CPU utilization and memory utilization of m VMs are collected during p time slices. Now the utilization matrix of h_i is calculated as the ratio of CPU utilization to memory utilization of m VMs on hi at each time slice.

$$RCM\_util_i[m][p] = \begin{bmatrix} RCM_{11} & RCM_{12} & RCM_{13} & \cdots & RCM_{1p} \\ RCM_{21} & RCM_{22} & RCM_{23} & \cdots & RCM_{2p} \\ \vdots & \vdots & RCM_{jk} & \ddots & \vdots \\ RCM_{m1} & RCM_{m2} & RCM_{m3} & \cdots & RCM_{mp} \end{bmatrix} \tag{3.7}$$

$$RCM_{jk} = \frac{CPU\_util_{jk}}{Mem\_util_{jk}} \tag{3.8}$$

where $[RCM_{jk}]$ refers to the ratio of CPU utilization to memory utilization of VM j on host h_i during time slice k

3. Product of CPU utilization to Memory utilization Algorithm (PCM) PCM method considers both CPU resource and memory resource. For host h_i with m VMs. The CPU utilization and memory utilization of m VMs are collected during p time slices. Now the utilization matrix of h_i is calculated as the product of CPU utilization to memory utilization of m VMs on h_i at each time slice.

$$PCM\_util_i[m][p] = \begin{bmatrix} PCM_{11} & PCM_{12} & PCM_{13} & \cdots & PCM_{1p} \\ PCM_{21} & PCM_{22} & PCM_{23} & \cdots & PCM_{2p} \\ \vdots & \vdots & PCM_{jk} & \ddots & \vdots \\ PCM_{m1} & PCM_{m2} & PCM_{m3} & \cdots & PCM_{mp} \end{bmatrix} \tag{3.9}$$

$$PCM_{jk} = CPU\_util_{jk} \times Mem\_util_{jk} \tag{3.10}$$

4. Experimental Setup. There are many difficulties that faces during testing and experimentation of Cloud Computing like demand for energy-efficient for IT technologies, demand time saving, and controlling the evaluation of algorithms, applications, and policies before real cloud products. One of the suitable approaches to make all these difficulties as easy is the simulations tools. The objective of this simulation tool is to offer an extensible framework that enables simulation, modeling, experimentation of Cloud computing infrastructures and application services. For this reason, simulation has been chosen to evaluate the performance of the algorithms.

4.1. Cloudsim Toolkit. The implementation was done using Cloudsim Toolkit [14]. The toolkit has been developed by the Cloud Computing and Distributed Systems (CLOUDS) Laboratory, University of Melbourne. CloudSim is completely written in Java. Netbeans or Eclipse IDE is used to run Cloudsim Toolkit. The simulation platform supports for modeling and simulation of large-scale Cloud computing data centers, virtualized server hosts, with customizable policies for provisioning host resources to virtual machines, energy-aware computational resources. And support for user-defined policies for allocation of hosts to virtual machines and policies for allocation of host resources to virtual machines. The data center is set up with 800 heterogeneous

hosts (physical nodes), half of which consists of HP Proliant ML 110 G4 and another half of which consists of HP Proliant ML 110 G5. The virtual machine's characteristics are corresponding to Amazon EC2. There are four types of virtual machines are considered such as High-CPU Medium instance, Extra-Large instance, Small instance, and Micro instance. The real-time workload data traces are used for this experiment as a part of the CoMon project Monitoring infrastructure for PlanetLab. The workload data used in the CPU utilization were taken from more than 500 places around the world [15]. In this experiment, 1516 VMs are chosen from '22/March/2011' dataset in workload traces.

4.2. Energy Consumption Model. A non-profit corporation called Standard Performance Evaluation Corporation (SPEC) is formed to establish, maintain and endorse standardized benchmarks and tools to evaluate performance and energy efficiency for the newest generation of computing systems [16] and [17]. All real data of energy consumption are derived from SPEC power benchmark. The different workload levels of energy consumption in the hosts are shown in Table 4.1.

TABLE 4.1
Host Energy Consumption in Different WorkLoad Levels

Server	HP G4	HP G5
0%	86	93.7
10%	89.4	97
20%	92.6	101
30%	96	105
40%	99.5	110
50%	102	116
60%	106	121
70%	108	125
80%	112	129
90%	114	133
100%	117	135

4.3. Evaluation Metrics. The following evaluation metrics are considered to compare the efficiency of proposed and existing methods.

4.3.1. Energy Consumption. The energy consumption by physical nodes in data center is mostly measured by CPU, memory, and network interfaces. Compared to other computing resources, the CPU consumes more energy. Also, most of these analyses have shown that an idle server consumes almost 70% of energy. This evidence legitimizes the method of converting the idle server to the power saver mode to reduce energy consumption. The energy model [18] is defined as, where E_{max} is the maximum energy consumed by a fully utilized server; k is the fraction of energy consumed by idle hosts (i.e., 70%), and CPU_Host is the host's CPU utilization. In our experiment, E_{max} value is set as 250W which is a constant value for modernized servers.

Due to the workload uncertainty, the CPU utilization may change over time and is defined as CPU_Host(t). Thus, the energy consumption by a host EC can be illustrated as an integral of power consumption over a while.

4.3.2. SLA Violation Rate. The SLA violation rate is one of the factors of QoS. It occurs while migrating VMs from overloaded host. It is based on two metrics [5] such as SVTAH and PDCVM.

1. SLA Violation Time per Active Host (SVTAH) SVTAH decides which active host has reached the 100% CPU utilization during the time. It is given in Eqn. 4.1,

$$SVTAH = \frac{1}{M}\sum_{i=1}^{M}\frac{T_{pi}}{T_{qi}} \tag{4.1}$$

where M is the number of hosts in the data center; T_{pi} is total time during which the i^{th} host reaches 100% CPU utilization; T_{qi} is total time of host i being in an active state.

2. Performance Degradation Caused by VM Migration (PDCVM) The overall performance will be degraded due to the VM migrations most of the time. It is formulated as

$$PDCVM = \frac{1}{N}\sum_{j=1}^{N}\frac{P_{dj}}{P_{ri}} \tag{4.2}$$

where N denotes the number of VMs; P_{dj} shows the estimation of performance degradation caused by migration of j^{th} VM; P_{ri} shows the lifetime of the total CPU capacity requested by j^{th} VM.
Both the SLA metrics are equally used to measure the SLA violation independently. It is obtained by multiplying SVTAH and PDCVM which is given Eqn. 4.3.

$$SLAV = SVTAH \times PDCVM \tag{4.3}$$

3. Energy SLA Violation (ESV) ESV is a combined metric that captures both energy consumption and SLA Violation rate, which are denoted as Energy Consumption (EC) and SLA Violation rate (SLAV).

$$ESV = EC \times SLAV \tag{4.4}$$

4. Energy Efficiency (EE) Energy Efficiency (EE) can be incorporated into forms of Energy Consumption and SLA Violation rate. Where EC is energy consumption. It is formed as

$$EE = \frac{1}{P_c \times SLA} \tag{4.5}$$

5. Improvement Rate The percentage improvement of the proposed algorithm is computed using the following Eqn. 4.6.

$$\varphi = \left(1 - \frac{Proposed\ Method}{Existing\ Method}\right) \times 100 \tag{4.6}$$

5. Results and Discussions. The proposed CPU-Memory aware placement algorithms are compared with following existing algorithms: PABFD [5], and MCC[10]. The parameter d varies from 0.6 to 1.0 by increase of 0.1 [5]. For $d < 0$ there is no CPU utilization of VMs and $d > 1$ there is no variation in these objectives. Hence the value of d is considered between 0.6 to 0.1 for all the algorithms. The impact of the proposed algorithm is experimentally tested with various selection algorithms and the obtained results are tabulated (Table 5.1).

The maximum efficiency is obtained when the correlation is based on memory utilization of hosts and VMs and it also takes less number of VM migrations for consolidation. The performance of the proposed method is discussed with respect to each metric as given below:

5.1. Energy Consumption. The main objective of this paper is to design a VM placement algorithm so that the energy consumption is reduced. Energy consumption is calculated by taking into all hosts throughout the simulation by mapping of CPU and different workload levels from Table 4.1. In every iteration the CPU utilization is measured and energy consumption is calculated from Table 4.1. The results obtained for existing and proposed methods are shown in Fig 5.1 and 5.2. From the figures it is observed that THR_FSS_MEM gives the minimum energy consumption than others. The proposed memory utilization based placement algorithm consumes less energy than other algorithms. It reduces energy consumption by 9.52 % than MCC and 1 % than PABFD. The minimum amount of energy consumed by the proposed algorithm is 12.16 KWh whereas the minimum energy consumption among the existing algorithms 12.28 KWh. Moreover the minimum energy consumption obtained by memory is based on the FSS selection algorithm as shown in Fig 3. The existing selection algorithms generate competitive results with FSS when they are combined with any of the proposed algorithms.

TABLE 5.1
Comparison of various Placement Algorithms

Algorithms	Energy Efficiency	Energy Consumption (KWh)	SLA Violation Rate ($\times 10^{-4}$)	Number of VM migrations
THR_RS_PABFD_0.6	180.48	18.3	3.03	2010
THR_MC_PABFD_0.7	124.29	27	2.98	3644
THR_MMT_PABFD_0.6	254.18	35.19	1.12	4899
THR_MU_PABFD_0.9	120.99	32.93	2.51	4597
THR_MP_MCC_0.6	240.26	38.29	1.09	2857
THR_RS_MCC_0.6	213.75	14.1	3.32	1774
THR_MC_MCC_0.7	236.69	15.7	2.69	1856
THR_MMT_MCC_0.6	164.13	14.1	4.32	1457
THR_MU_MCC_0.6	168.95	14.89	3.98	1588
THR_RS_MEM_0.7	161.77	13.55	4.56	1649
THR_RS_RCM_1.0	490.97	16	1.27	1837
THR_RS_PCM_0.9	166.11	17.8	3.38	2077
THR_MC_MEM_0.6	179.43	14.39	3.87	1666
THR_MC_RCM_0.7	239.8	17.7	2.36	1760
THR_MC_PCM_0.9	197.02	18.39	2.76	1567
THR_MMT_MEM_0.6	239.93	16.37	2.55	1631
THR_MMT_RCM_0.7	183.57	13.68	3.98	1418
THR_MMT_PCM_1.0	230.84	14.77	2.93	1735
THR_MU_MEM_0.9	130.26	17.09	4.49	1782
THR_MU_RCM_0.6	185.67	12.83	4.20	1670
THR_MU_PCM_1.0	122.38	12.16	6.72	1374
THR_FSS_MCC_0.7	680.12	13.44	1.09	1564
THR_FSS_PABFD_0.7	318.6	12.28	2.56	1670
THR_FSS_MEM_0.6	704.79	12.49	1.14	1257
THR_FSS_RCM_0.7	594.35	15.55	1.08	1625
THR_FSS_PCM_0.7	485.84	18.56	1.11	1629

5.2. SLA Violation Rate. SLA violation is one of the key factors of Quality of Service (QoS). It is calculated by taking into two scenarios. First thing is to find overloaded host detection and the next is incurred for migration. K-Means Inter Quartile Range clustering algorithm [7] efficiently finds out the overloaded host. If host overload is predicted efficiently then there will be fewer migrations which will reduce the SLA violation rate. The SLA violation rate is also based on the number of VM migrations. If number of violations are less then SLA violation rate will be less. The proposed RCM (FSS_RCM_0.7) based placement algorithm obtains minimum SLA violation rate as it gets less number of VM migrations. The obtained experimental results are shown in Fig 5.3 and 5.4. It reduces SLA violation by 1 % than MCC and 3.4 % than PABFD. Moreover FSS selection algorithm outperforms other algorithms in terms of generating minimum SLA violation rate. Fig 5.4. shows that the existing selection algorithms reduce the SLA violation rate when they are combined with the proposed placement algorithms.

5.3. Energy Efficiency. The energy efficiency is inversely proportional to Energy Consumption and SLA violation rate. The algorithm which gets minimum EC and SLA violation rate will get minimum energy efficiency. Fig 5.5. shows the energy efficiency obtained by various placement algorithms. It is observed that the FSS based memory aware placement algorithm maximizes the energy efficiency. It outperforms MCC by 3.62 % and PABFD by 121.21 % in terms of improving energy efficiency. The maximum energy efficiency obtained by FSS_MEM_0.6 is 704.79 and the maximum among other algorithms is 680.12. Hence, the FSS_MEM VM placement algorithm is the most energy efficient for VM placement.

5.4. Number of VM Migrations. Less number of VM migrations means efficient VM consolidation and minimum SLA violation rate. Since the proposed placement algorithm uses five thresholds to classify the data

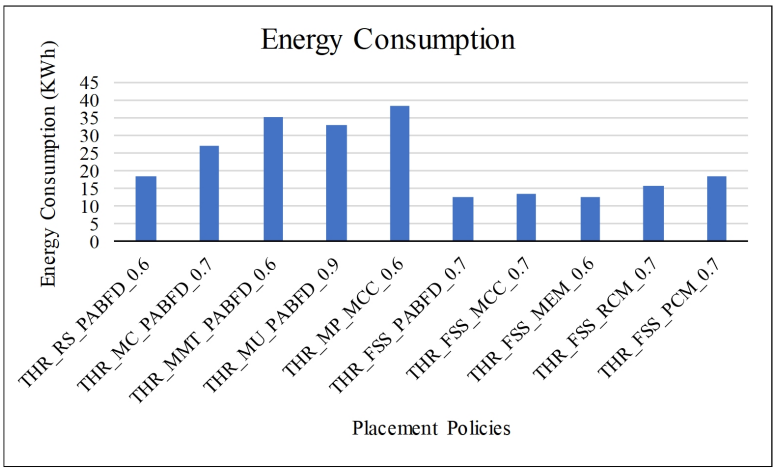

FIG. 5.1. *Comparison of Energy Consumption using various VM placement Algorithms*

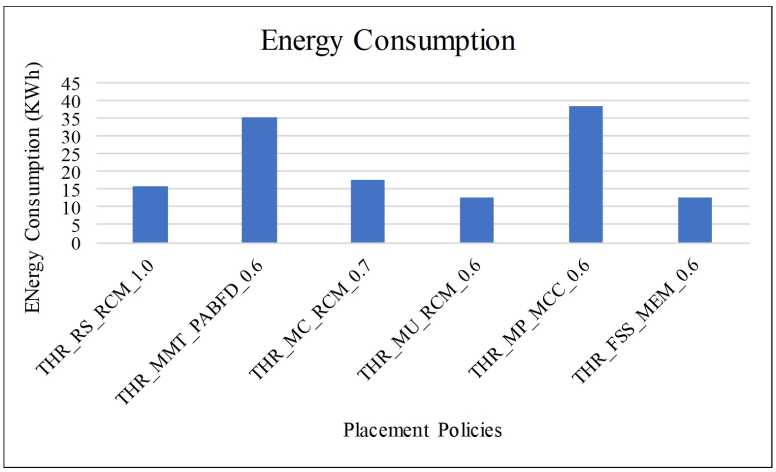

FIG. 5.2. *Energy Consumption using VM placement Algorithms*

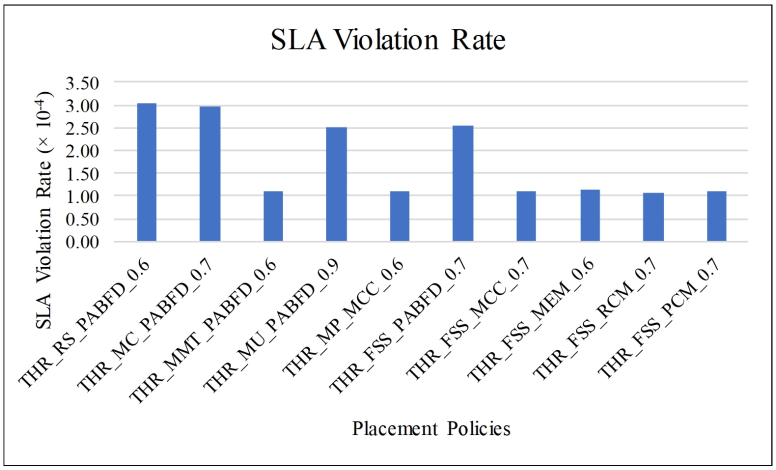

FIG. 5.3. *Comparison of SLA Violation Rate using various VM placement Algorithms*

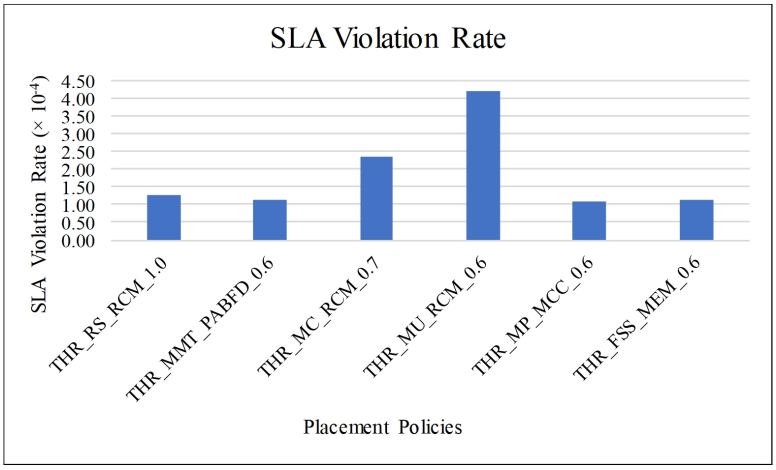

FIG. 5.4. *SLA Violation Rate using various VM placement Algorithms*

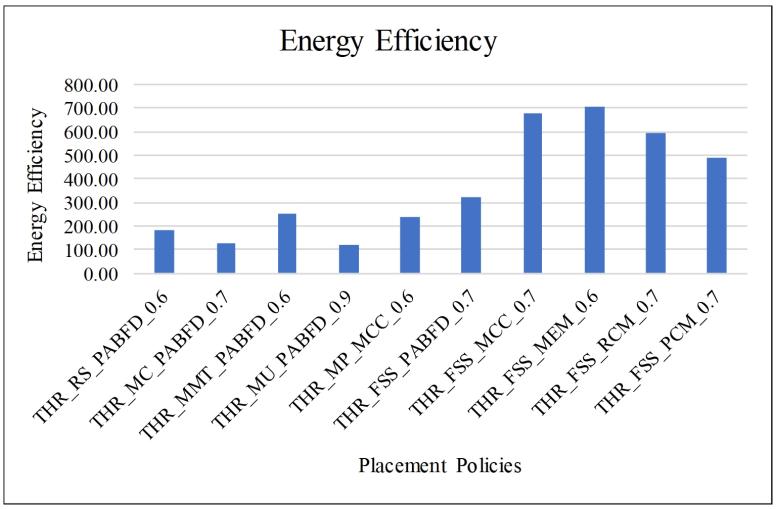

FIG. 5.5. *Comparison of Energy Efficiency using Various VM Placement Algorithms*

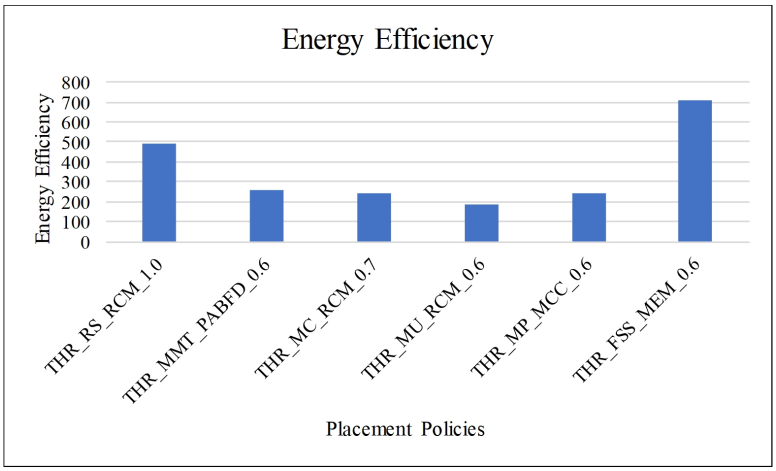

FIG. 5.6. *Energy Efficiency using various placement Algorithms*

center hosts into five clusters of hosts and applies fuzzy soft set for VM selection it obtains minimum number of VM migrations. The number of VM migration caused by FSS_MEM 0.6 is 1257 whereas the minimum caused by other algorithms is 1457. FSS_MEM algorithm resulted in 13.73 % reduction in migration than the other existing VM placement algorithms.

5.5. Observations. From the simulation results the following observations are made:

1. The existing selection algorithms (RS, MC, MMT, MU, MP) used two thresholds while algorithm FSS used four thresholds for host classification. In our previous work [7], it is identified that the adaptive four threshold algorithm is more effective than two and three threshold algorithms due to better prediction of overloaded hosts and underutilized hosts.
2. The existing VM selection (RS, MC, MMT, MU, MP) algorithms have considered any one of the factors such as either RAM or CPU or Memory or Correlation values for selecting VMs for migration. But are FSS based VM selection algorithm takes into consideration all the four factors at the same time. From the results of [9] it is observed that the FSS algorithm accurately finds which VM has to be migrated from the overloaded hosts.
3. The existing VM placement algorithms (PADFD, MCC) are compared with proposed VM placement algorithms (MEM, RCM, PCM). During the VM placement, the algorithm FSS_MEM achieves maximum energy efficiency. It is experimentally proved that the latter has better performance than the formers.
4. The proposed FSS based VM placement algorithms outperform other selection and placement algorithms in terms of maximizing energy efficiency, minimizing energy consumption, minimizing SLA violation rate, and minimizing the number of VM migrations.

6. Conclusion. In this paper, the VM placement problem is addressed for improving the resource utilization across multiple dimensions with the goal of maximizing energy efficiency and minimizing SLA violation rate. Multiple resource-constraint factors, such as CPU utilization and Memory utilization are used to migrate VMs onto the appropriate hosts in cloud data centers. CPU-Memory aware VM placement algorithm is proposed which considers three variations of resource utilizations: Memory, Ratio of CPU to memory utilization (RCM), and Product of CPU and memory (PCM) utilization. The proposed algorithm is implemented for real-world dataset and the experimental results are compared with existing selection and placement algorithms for various metrics. The results show that THR_FSS_MEM outperforms energy efficiency by (3.62 %) than MCC and the (121.21%) than MCC. THR_MU_PCM outperforms energy consumption (9.52 %) than MCC and (1%) than PABFD. THR_FSS_RCM outperforms SLA violation rate (3.4 %) than PABFD and (1%) than MCC. THR_MEM_0.6 outperforms number of VM migration by (13.73 %) than PABFD and MCC.

Currently the serial processing with multiple iteration is used to process the VM placement methods. All the four methods give the best result. But it consumes more time during the implementation. The parallel processing of all the 4 VM placement methods like CPU, MEMORY, RCM, and PCM method together was a challenge. The above challenges will be overcome in future using GPU systems.

From the experiments and detailed analysis, the VM placements can be done using either MEM or PCM or RCM strategies with fuzzy soft set VM selection policy. They are giving competitive results in terms of generating quality of service during VM consolidation. In the future work, the machine learning or deep learning based prediction method will be applied to dynamically predict VM placement method.

REFERENCES

[1] Hao Jin, Deng Pan, Jing Xu, and Niki Pissinou, *Efficient vm placement with multiple deterministic and stochastic resources in data centers, IEEE Global Communications Conference (GLOBECOM)*, IEEE, 2012, pp. 2505–2510.
[2] Donald J. Daly and Donald J. Daly `https://aws.amazon.com/ec2/`Economics 2: Ec2, 1987.
[3] Code and response `https://www.ibm.com/us-en/`
[4] google cloud `https://cloud.google.com/products/`
[5] Anton Beloglazov and Rajkumar Buyya *Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers*, Concurrency and Computation: Practice and Experience, Wiley Online Library, 2012, pp. 1397–1420.
[6] Nithiya Baskaran and R Eswari *An Efficient Threshold-Fuzzy Based Algorithm for VM Consolidation in Cloud Datacenter*, 2020, International Journal of Grid and High Performance Computing, IGI, (In press).

[7] Nithiya Baskaran and R Eswari *Adaptive threshold-based algorithm for multi-objective vm placement in cloud data centers*, In International Conference on Frontier Computing, Springer, 2018, pp. 118–129.
[8] Zhou Zhou, Jemal Abawajy, Morshed Chowdhury, Zhigang Hu, Keqin Li, Hongbing Cheng, Abdulhameed A Alelaiwi, and Fangmin Li *Minimizing sla violation and power consumption in cloud data centers using adaptive energy-aware algorithms*, Future Generation Computer Systems, Elsevier, 2018, pp.836–850.
[9] Nithiya Baskaran and R Eswari *Fuzzy Softset Based VM Selection in Cloud Datacenter*, International Conference on Intelligent Computing and Control Systems [ICICCS 2019], IEEE Xplore Digital Library, IEEE, (In Press).
[10] Xiong Fu and Chen Zhou *Virtual machine selection and placement for dynamic consolidation in cloud computing environment*, Frontiers of Computer Science, Springer, pp.322–330, 2015.
[11] Alharbi, Fares and Tian, Yu-Chu and Tang, Maolin and Zhang, Wei-Zhe and Peng, Chen and Fei, Minrui *An ant colony system for energy-efficient dynamic virtual machine placement in data centers*, Expert Systems with Applications, Elsevier, 2019, pp.228–238.
[12] Al-Moalmi, Ammar and Luo, Juan and Salah, Ahmad and Li, Kenli *Optimal virtual machine placement based on grey wolf optimization*, Electronics, Multidisciplinary Digital Publishing Institute, 2019, pp.283.
[13] Parvizi, Elnaz and Rezvani, Mohammad Hossein *Utilization-aware energy-efficient virtual machine placement in cloud networks using NSGA-III meta-heuristic approach*, Cluster Computing, Springer, 2020, pp. 1–23.
[14] Rodrigo N Calheiros, Rajiv Ranjan, Anton Beloglazov, César AF DeRose, and Rajkumar Buyya, *Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms*, Software: Practice and experience, Wiley Online Library, 2011, pp.23–50.
[15] KyoungSoo Park and Vivek S Pai, *Comon: a mostly-scalable monitoring system for planetlab*, ACM SIGOPS Operating Systems Review, ACM, 2006, pp.65–74.
[16] Xiaobo Fan, Wolf-Dietrich Weber, and Luiz Andre Barroso, *Power provisioning for a warehouse-sized computer*, ACM SIGARCH computer architecture news, volume 35, ACM, 2007, pp. 13–23.
[17] Dara Kusic, Jeffrey O Kephart, James E Hanson, Nagarajan Kandasamy, and Guofei Jiang, *Power and performance management of virtualized computing environments via lookahead control*, Cluster computing, Springer, 2009, pp.1–15.
[18] Anton Beloglazov, Jemal Abawajy, and Rajkumar Buyya, *Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing*, Future generation computer systems, Elsevier, 2012, pp.755–768.

Edited by: P. Vijaya
Received: Dec 10, 2019
Accepted: Jun 10, 2020